Erzeugen einfacher OpenStreetMap-Karten

Chemnitzer Linux-Tage 2013

Jens Pönisch

2013-03-16

Inhalt

1. Überblick
2. Von der Kugel zur Fläche – Projektionen
3. Kacheln und Slippy-Map
4. Karten für den Druck
5. Online-Karten
6. Karten für Garmin-GPS-Empfänger
7. Karten für Android-Geräte
8. Quellen

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Warum eigene Karten erstellen?

Papierkarten:

- Stadtpläne
- Urlaubsziel
- Karten mit ganz bestimmten POIs
- Dokumentation von Touren/Reisen

Online-Karten:

- Lagepläne (Ort der Chemnitzer Linux-Tage)
- Anfahrtskizzen
- Dokumentation von Touren/Reisen

Karten für Garmin-Geräte:

- Kommerzielle Karten teuer
- Gewünschte Region nicht fertig verfügbar

Allgemein:

- Erfolgserlebnis
- Wissenszuwachs

OpenStreetMap

Was ist OpenStreetMap?

- 2004 von Steve Coast in UK gegründet
- Geodatenbank mit ca. 2 Mrd. geographischen Objekten
- Online- und Offline-Zugang
- Vektordaten
- Datenquellen: GPS-Geräte, Luftbilder, Datenspenden
- Netzwerk von ca. 200 000 aktiven Mitwirkenden: Datenerfasser und Werkzeugentwickler
- Editoren zum Datenerfassen
- Tools zur Datenverarbeitung

Was ist OpenStreetMap *nicht*?

- Kartenanbieter
- Anbieter von Navigationssystemen, -software
- Wünsch Dir was

Aber:

- Projekteigene Karten vorhanden.
- Anleitungen zur Datenverarbeitung, Erstellen eigener Karten.
- Viele Anwendungen zur Datenauswertung vorhanden:
  auch Kartenerstellung und Navigation
- Kommerzielle Dienstleister für OSM-Daten

Lizenz

- Open Datebase License
- Nutzung der Daten auch kommerziell möglich
- Bei Veröffentlichung Namensnennung erforderlich:
  Map Data from OpenStreetMap Contributors bzw.
  Kartendaten: OpenStreetMap Mitwirkende

Nutzung fertiger Karten auf OSM-Basis:

- Lizenz der Karte beachten
- meist CC-Lizenz, teilweise kommerzielle Nutzung ausgeschlossen

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

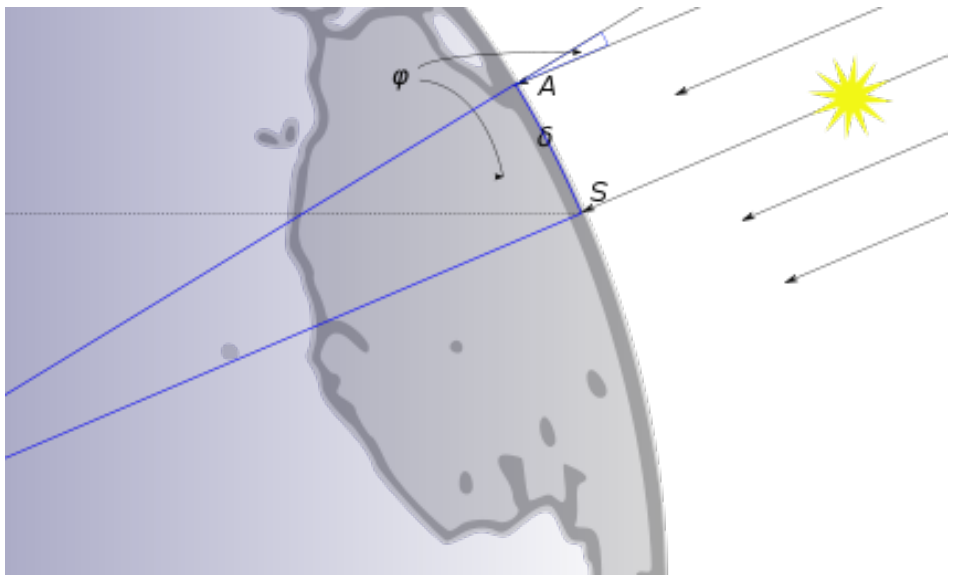
6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Die Erde ist eine Kugel

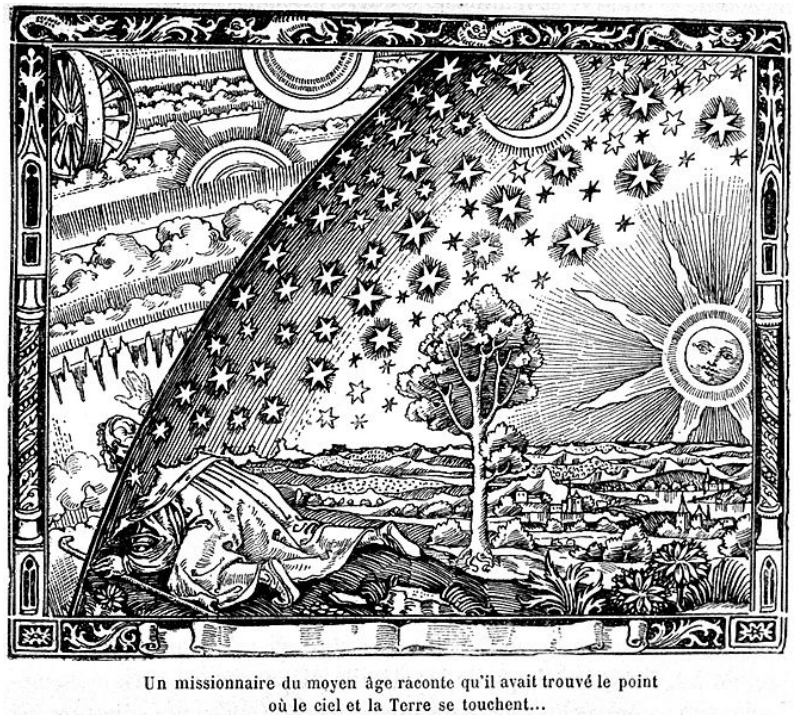
Seit mindestens 2500 Jahren bekannt.
Eratosthenes (200 v. Chr.): Berechnung des Erdumfangs.



Wikimedia Commons

Die Erde ist eine Scheibe

Mittelalter (6. bis 12. Jh.)



Holzstich von Fammarion, 16. Jh. (Wikimedia Commons)

Kosmas Indikopleustes: *Topographia Christiana*, 550 n. Chr.
Keine Mehrheitsmeinung!

Die Erde ist ein Quadrat

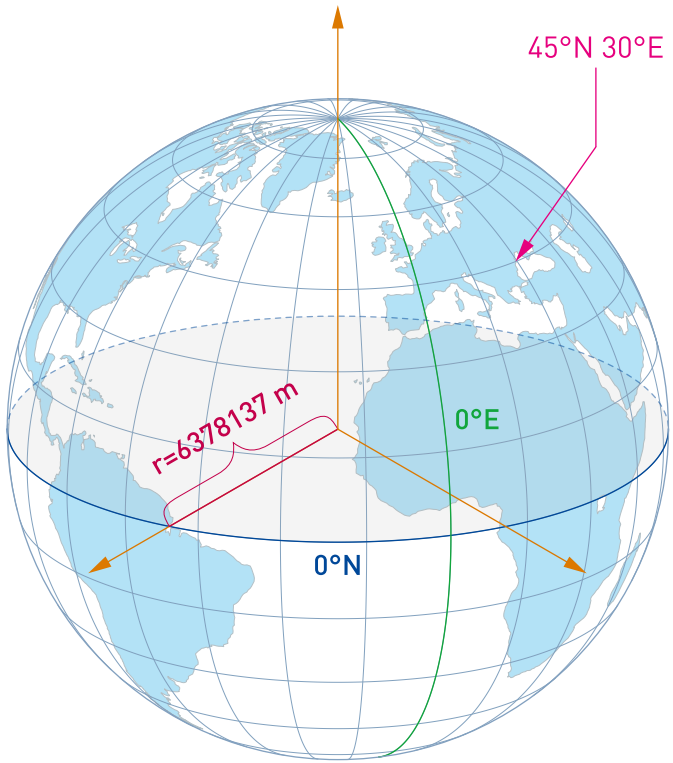
Google Maps, 2005

OpenStreetMap Contributors

Grundidee: 1569 von *Gerhard Mercator*
Wie wird aus der Kugel ein Quadrat?

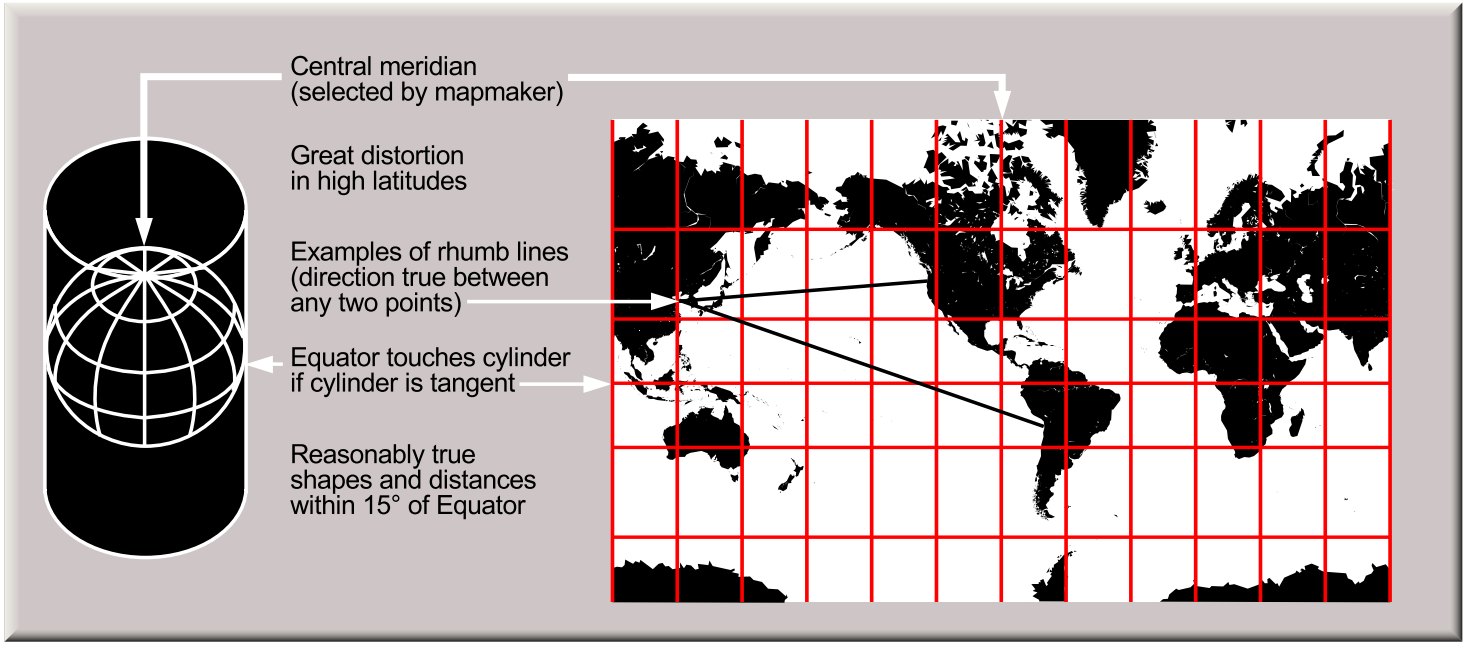
Positionsangaben auf der Erde – Gradnetz

- Breitengrade (Latitude): parallel zum Äquator
  Numerierung bis 90° nach Nord bzw. Süd
- Längengrade (Longitude): Großkreise durch die Pole
  Numerierung bis 180° nach Ost bzw. West
  Längengrad 0: Greenwich
- Südliche Breiten und westliche Längen: negatives Vorzeichen
- Referenzsystem: WGS 84



Quelle: René Schwarz (CC-BY-NC-SA 3.0)

Mercator-Projektion 1



Wikimedia Commons

- Zylinder in Nord-Süd-Richtung über die Erde stülpen
- Strahlen vom Erdmittelpunkt durch die Erdoberfläche auf Zylinder
- Zylinder aufschneiden und flach ausbreiten
- geeignet in Nord-Süd-Richtung strecken

Mercator-Projektion 2

Fläche verzerren:

- Himmelsrichtungen erhalten
- Form kleiner Gebiete erhalten

$$x = s \cdot \lambda$$

$$y = s \cdot \ln\left(\tan\left(\frac{\pi}{4} + \frac{\varphi}{2}\right)\right)$$

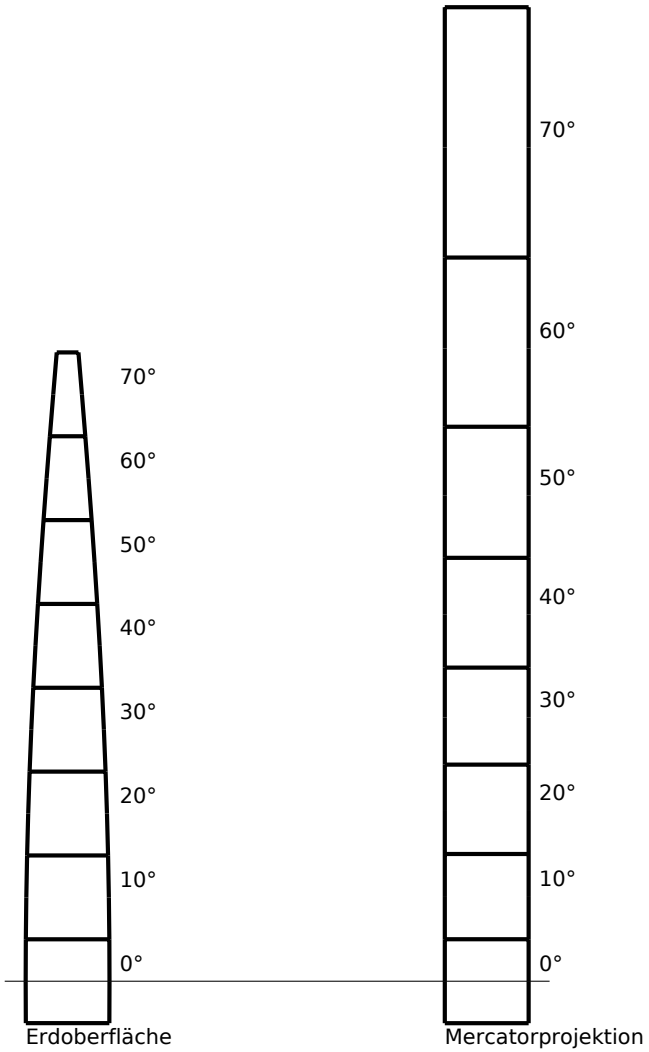
$$= s \cdot \ln\left(\tan\varphi + \frac{1}{\cos\varphi}\right)$$

λ – Längengrad in Radiant

φ – Breitengrad in Radiant

s – Skalierungsfaktor

$$\text{Radiant} = \frac{\pi}{180^0} \cdot \text{Grad}$$



Mercatorprojektion – Eigenschaften

- winkel- und richtungstreu
- Form und Lage kleiner Regionen bleibt erhalten
- nicht flächentreu
- Maßstab vom Breitengrad abhängig
- Darstellung nur bis zum 82-ten Breitengrad möglich.
  Aber:
  - Anchorage: 61°N
  - Nordkapp: 71°N
  - Longyearbyen (Spitzbergen): 78°N
  - Navarino (Feuerland): 55°S

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

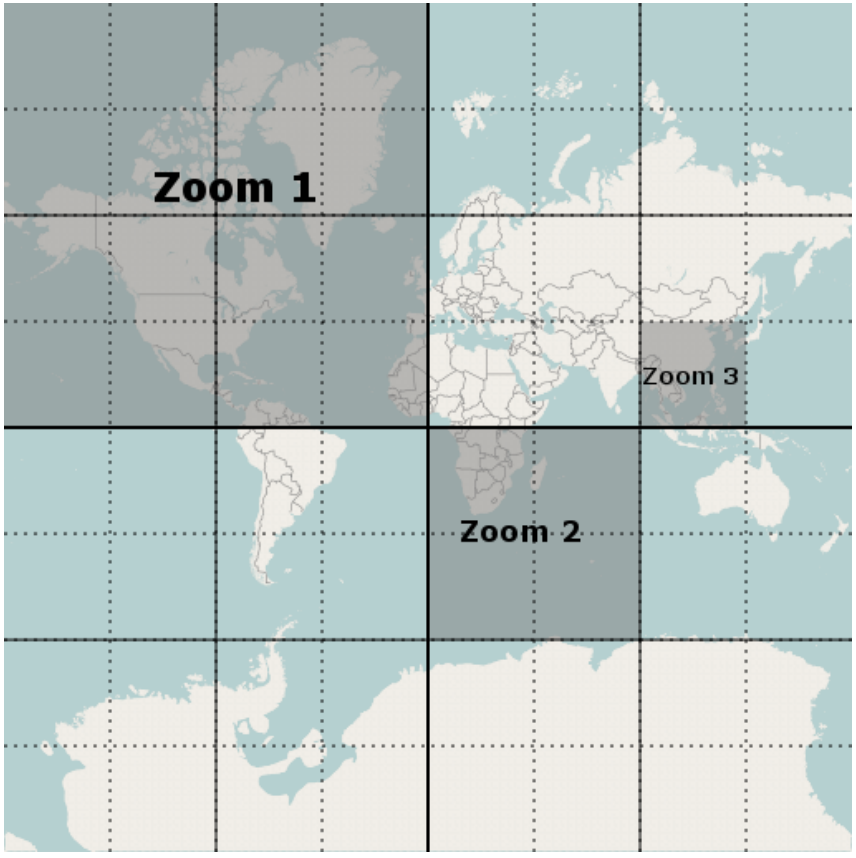
6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Slippy-Map

- Zerlegen der Karte in Kacheln der Größe 256×256 Pixel
- Zoomstufe 0: 1 Kachel
- nächste Zoomstufe: Verdoppelung der Kachelzahl in x- und y-Richtung
- Kachelbeschreibung: (z, x, y)
- Ursprung – Kachel (0,0) – linke obere Ecke



OpenStreetMap Contributors

Zoomstufen und Maßstab

- Kacheln meist von Zoomstufe 0 bis 16 oder 18
- Zoomstufen bis 10 sehr grob, meist anderer Kartenstil
- Maßstab bei Kachelgröße 4×4 cm^2:

Zoom	Kachelgröße[km]	Maßstab
11	12,5	1:300 000
12	6	1:150 000
13	3	1:75 000
14	1,5	1:38 000
15	0,8	1:19 000
16	0,4	1:10 000

Berechnung der Kachelnummer

- Geographische Koordinaten ermitteln (z. B. aus Online-Karte)
- Auswahl der gewünschten Zoomstufe z.
- x- und y-Koordinate der Kachel aus Mercator-Projektion, Zoomstufe z bestimmt Skalierung s:

$$\lambda = \frac{\pi}{180} \cdot \text{Längengrad} \qquad \varphi = \frac{\pi}{180} \cdot \text{Breitengrad}$$

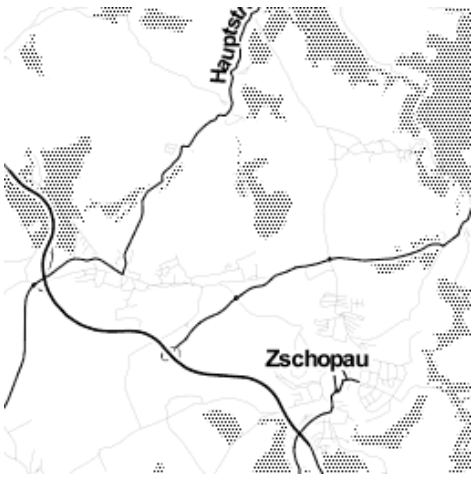
$$x = \left\lfloor 2^z \cdot \frac{\lambda + \pi}{2 \cdot \pi} \right\rfloor \qquad y = \left\lfloor 2^z \cdot \frac{\pi - \ln(\tan\varphi + \frac{1}{\cos\varphi})}{2 \cdot \pi} \right\rfloor$$

- Programmcode für viele Programmiersprachen im OSM-Wiki

Basiskarte und Overlays

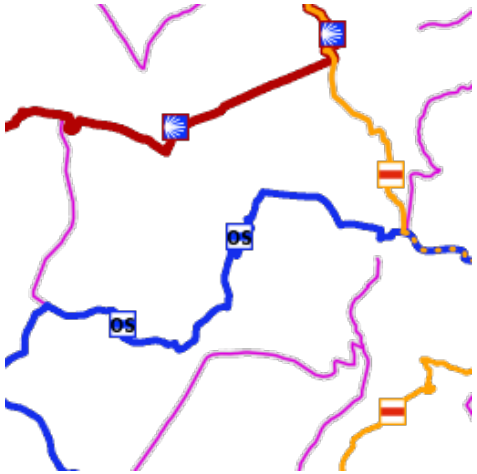
Basiskarten

Mapnik



Toner

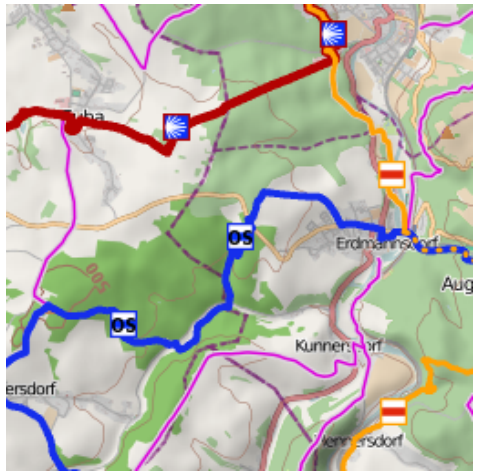
Overlays

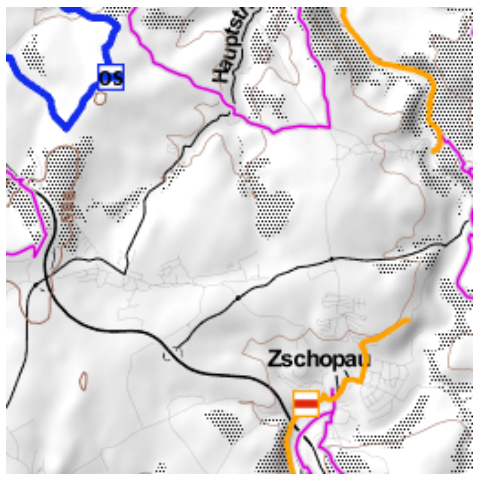


Lonvia-Wanderwege

Höhenlinien

Kombination





Datenquellen für Kacheln

- Bereitstellung per Webserver
- Aufbau der URLs:
  http://server/pfad/**z**/**x**/**y**.png

Problem: Ermitteln von Server und Pfad einer Kachelquelle.

Ermitteln von Kachelquellen

URL der Datenquelle ist in der Kartenanwendung eingebaut.
Der WWW-Browser «kennt» diese.

- Anzeige der gewünschten Online-Karte im Browser
- Funktion *Graphik anzeigen* zeigt Kachel-URL:
  kann gesperrt sein, funktioniert nicht bei Overlays
- Analyse des Quellcodes der Kartenanwendung:
  aufwendig
- Mitscannen des Datenverkehrs:
  - Wireshark etc.:
    benötigt Root-Rechte, Filter setzen, recht aufwendig
  - Firefox-Plug-in *Live HTTP headers*: Anzeige aller URLs, die von der Anwendung aufgerufen werden, enthält alle Kachel-URLs
- Demo *Live HTTP headers*

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Prinzipielles Vorgehen

Vorbereitungen:

- Festlegen der gewünschten Region
- Festlegen der Zoomstufe
- Auswahl des Kacheltyps / der Datenquelle

Prozess:

- Download der benötigten Kacheln als Bilddateien
- Zusammensetzen der Kacheln zum Gesamtbild
- Ausdruck in der gewünschten Größe

Werkzeuge

Download	cURL und wget
Montage	ImageMagick, Gimp
Ausdruck	Bildbetrachter, Gimp, evt. Auftrennen in mehrere Bilder mit ImageMagick

Fazit: Manuell sehr mühsam.

Hilfsmittel

Automatisieren von Download und Zusammensetzen mittels Script:

bigmap.cgi Online-Programm, Nutzung mühsam

bigmap.py Python-Script, lokale Nutzung, leicht anpassbar.
Benötigt *cairo*-Bibliothek
(Debian-Paket: python-cairo)

Nutzung: `python bigmap.py -?`

```
Usage: python bigmap.py options
Options:
-b | --bbox lon0,lat0,lon1,lat1     Bounding box
-t | --tiles x0,y0,x1,y1            Alternative: tile numbers
-z | --zoom zoomlevel               Tile's zoom level
-f | --out file                     Image file name
-s | --style style                  Map style

Styles:
 * cyclemap
 * watercolor
 ...
```

Beispiel bigmap.py

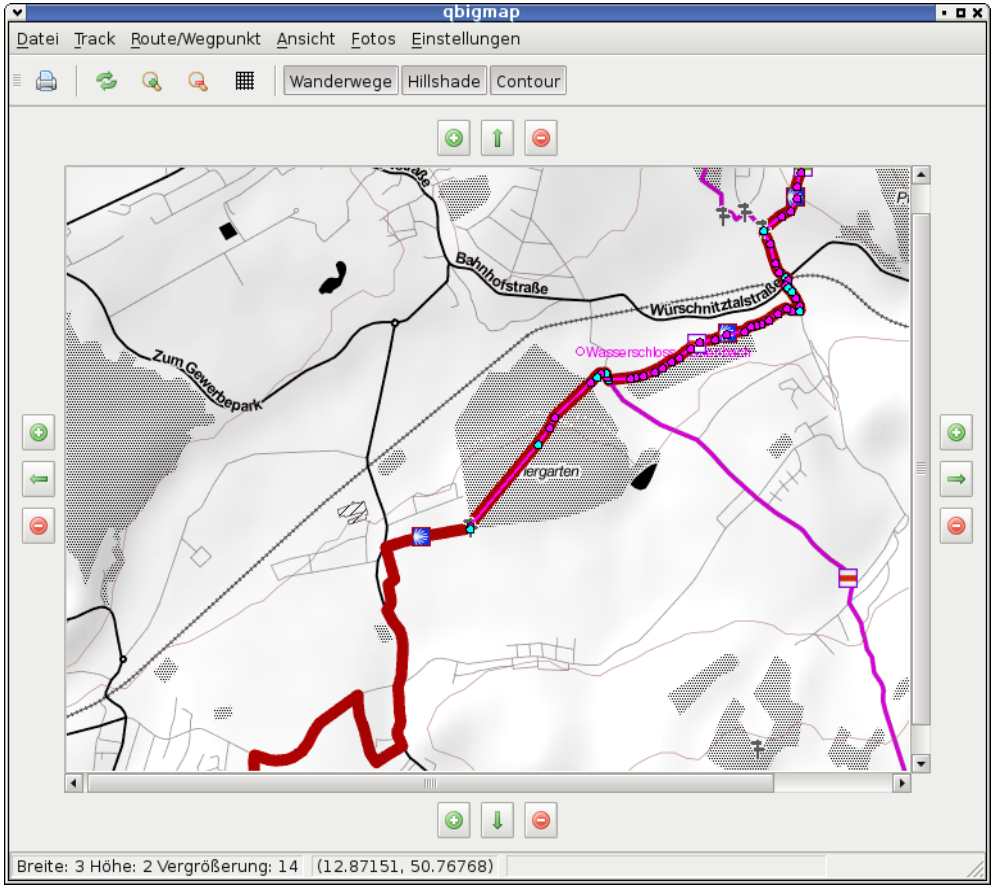
```
python bigmap.py -b 12.90,50.81,12.95,50.82 -z 14 -s osb
```

QBigmap

GUI-Interface zur Auswahl des Kartenausschnitts und zum Zusammensetzen von Karten.
Weitere Möglichkeiten:

- Overlays
- Darstellung von GPX-Tracks
- Darstellung von POIs (GPX-Waypoints)
- Entwurf von GPX-Routen
- Erzeugen von Online-Karten (s. u.)

Demo

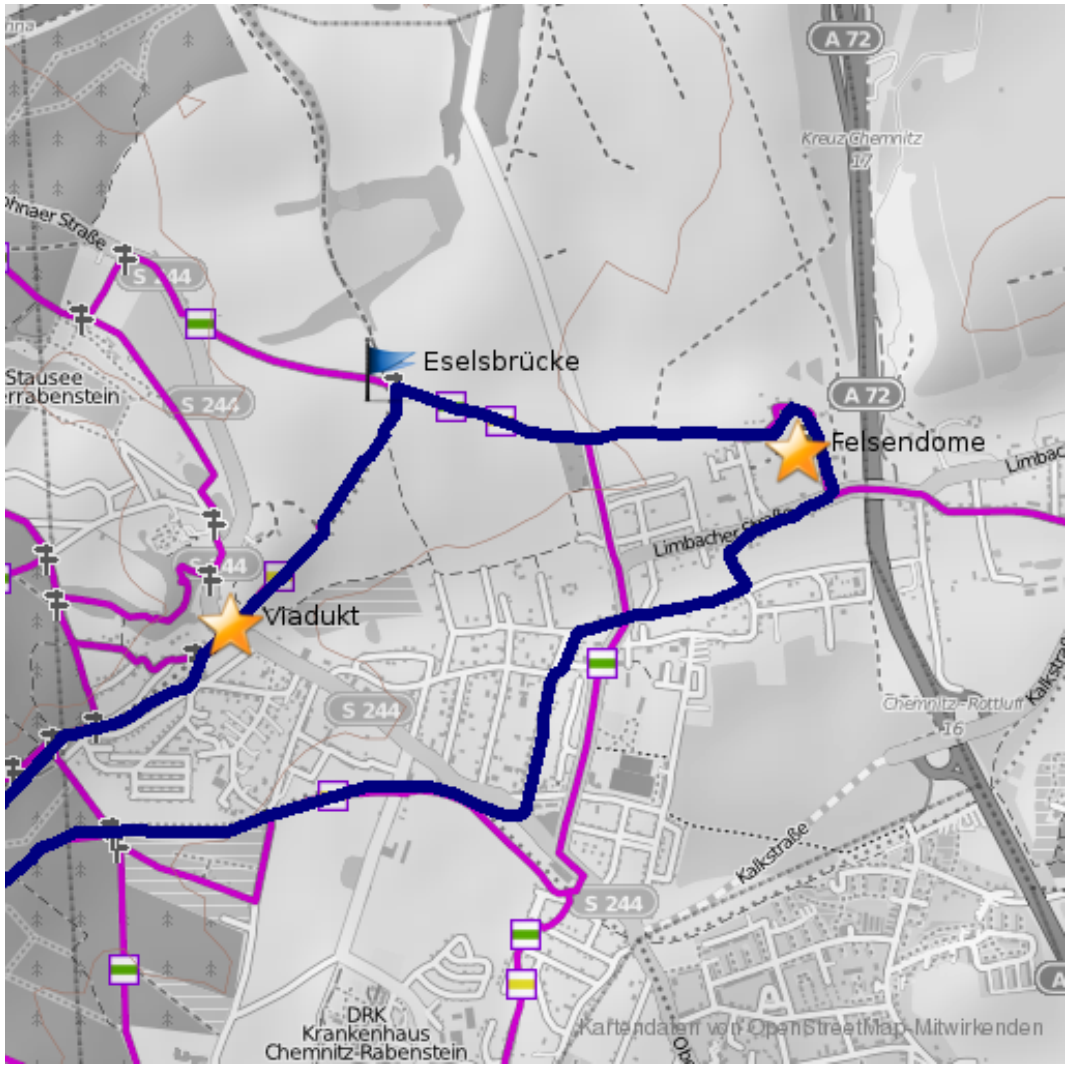


Ergebnis QBigmap

Style: Black-White
Overlays:

- Lonvia-Wanderwege
- Höhenschattierung
- Höhenlinien

Track und POIs
Rabenstein



1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Aufbau

- Kacheln werden mittels JavaScript-Bibliothek geladen und zusammengesetzt.
- Kachelquelle benötigt
- Anwendung besteht aus mindestens 3 Teilen:
  - HTML-Datei
  - JavaScript-Bibliothek
  - eigener JavaScript-Code
- Test ohne eigenen Webserver lokal möglich

Einbinden weiterer Daten:

- GPX-Tracks
- POIs
- ...

JavaScript-Bibliotheken für Online-Karten

OpenLayers	nicht nur für OpenStreetMap geeignet
	sehr umfangreich und flexibel
	relativ kompliziert in der Nutzung
Leaflet	neuere Bibliothek der Firma Cloudmade
	speziell für OpenStreetMap
	einfacher Einstieg

Beide Bibliotheken sind frei nutzbar.
Hier: *Leaflet*

HTML-Seite

Grundaufbau der Kartenseite, JavaScript-Code in eigener Datei.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Leaflet-Karte 1</title>
<!-- Leaflet-Symbole -->
<link rel="stylesheet"
     href="http://cdn.leafletjs.com/leaflet-0.4/leaflet.css"/>
<!-- Leaflet-Bibliothek -->
<script src="http://cdn.leafletjs.com/leaflet-0.4/leaflet.js">
</script>
<!-- Eigene Kartenfunktionalität -->
<script src="myleaf1.js"></script>
</head>

<!-- Initialisierung mit Breite, Länge, Zoom -->
<body onload="javascript:map(50.8,12.9,12)">
<!-- Kartenfläche, Größe muss gesetzt sein -->
<div id="map" style="width:100%;height:100%;"></div>
</body>
```

JavaScript-Datei

Datei `myleaf1.js` im gleichen Verzeichnis:

```
function map(lat, lon, zoom) {
  // Kartenobjekt mit Position und Zoomfaktor erzeugen
  var myMap = L.map('map').setView([lat, lon], zoom);
  // Attributionstring (Copyright) setzen
  var attribution = 'Map data &copy; '
    +'<a href="http://openstreetmap.org">'
    +'OpenStreetMap</a> contributors, '
    +'<a href="http://opendatacommons.org/licenses/odbl/">ODbL</a>';
  // Kartenlayer hinzufügen, Platzhalter {z}, {x}, {y}
  L.tileLayer('http://tile.openstreetmap.org/{z}/{x}/{y}.png',
  { attribution: attribution,
    maxZoom: 18
  }).addTo(myMap);
}
```

Demo

Anzeige von POIs

Egänzen eines *Marker*-Objekts

Pop-up zeigt angegebenen HTML-Inhalt an

Bild im gleichen Verzeichnis ablegen

```
function map(lat, lon, zoom) {
  var myMap = L.map('map').setView([lat, lon], zoom);
  var attribution = 'Map data &copy; '
    +'<a href="http://openstreetmap.org">'
    +'OpenStreetMap</a> contributors, '
    +'<a href="http://opendatacommons.org/licenses/odbl/">ODbL</a>';
  L.tileLayer('http://tile.openstreetmap.org/{z}/{x}/{y}.png',
  {
    attribution: attribution,
    maxZoom: 18
  }).addTo(myMap);

  // Marker setzen
  var marker = L.marker([50.8135, 12.9293]).addTo(myMap);
  // Beim Anklicken angezeigter Inhalt:
  marker.bindPopup(
    '<b>Chemnitzer Linux-Tage<br/><img src="wahl.png"/>'
  );
}
```

Anzeige eines GPX-Tracks

Problem: Track eigene Datei, muss geladen und verarbeitet werden.
Grundfunktion Polygondarstellung in Leaflet vorhanden.

```
function map(lat, lon, zoom) {
  ...
  drawGpx(myMap, 'mytrack.gpx');
}
function drawGpx(map, gpxfile) {
  // Download Track
  var xmlhttp = new XMLHttpRequest();
  xmlhttp.open('GET', gpxfile, true);
  xmlhttp.overrideMimeType('application/xml');
  xmlhttp.onreadystatechange = function () {
    if (xmlhttp.readyState != 4) return;
    if (xmlhttp.status != 0 && xmlhttp.status != 200) return;
    // Download ok - Verarbeitung
    // Für Route rtept, für Wegpunkte wpt verwenden
    var nodes = xmlhttp.responseXML.getElementsByTagName('trkpt');
    drawNodes(map, nodes);
  }
  xmlhttp.send();
}
```

GPX-Track (Fortsetzung)

```
function drawNodes(map, nodes) {
  poly = [];
  for (var i = 0; i < nodes.length; i++) {
    var lat = parseFloat(nodes[i].getAttribute('lat'));
    var lon = parseFloat(nodes[i].getAttribute('lon'));
    poly.push(new L.LatLng(lat, lon));
  }
  // Track darstellen
  var track = new L.Polyline(poly, {
    color: '#ff00ff',
    weight: 3,
    opacity: 1.0,
  }).addTo(map);
  // Bei Anklicken des Tracks anzeigen
  track.bindPopup("Trackbeschreibung");
}
```

Weitere Funktionen von Leaflet

- Auswahl der Basiskarte
- eigene Popupmarker
- Kilometerskala
- Flächendarstellungen
- Ein- und Ausblenden von Overlays
- ...

Hinweis: QBigmap erzeugt derzeit OpenLayers-Karten

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Garmin-Karten

- Voraussetzung: Garmin-Gerät kann Karten darstellen und externe Karten laden.
- Datenformat nicht offengelegt, durch Reverse Engineering ermittelt.
- Vektor- und Rasterdaten – wir betrachten nur Vektordaten.
- Karte üblicherweise auf SD-Karte, Größenbeschränkung bei älteren Geräten.
- Großes Angebot fertiger Karten, oft auf OSM-Basis
- Online-Karten-Generator $\Rightarrow$ OSM-Stand

Kartengenerierung nicht schwierig.
Java-Laufzeitsystem benötigt.

Datenquellen

OpenStreetMap-Vektordaten, keine Kacheln, benötigt.
Download der Daten (OSM-XML- oder PBF-Format):

- Planet Dump:
  `http://planet.openstreetmap.org/planet/`
  (Größe: 30 GB)
- Kontinent- und Länderauszüge:
  `http://download.geofabrik.de/openstreetmap/`
  (Größe Europa: 10 GB, Deutschland: 1.5 GB)

Online-Abfrage (kleine Regionen bis $100 \times 100\,\mathrm{km}^2$):

- `http://overpass-api.de/api/`
- Mirror (s. OSM-Wiki)
- eigener Overpass-API-Server

Benötigte Werkzeuge

Osmosis	Zuschneiden der benötigten Kartenregion
Splitter	Zerlegen der Kartenregion in Garmin-Kacheln
Mkgmap	Generieren der Garmin-Karte
Style und Typfiles	Auswahl der darzustellenden Kartenelemente, Festlegen der Darstellung

Style und Typ: Files von *Computerteddy*
im Garmin-Arbeitsverzeichnis ablegen

Auswahl der benötigten Region

Datendump: Werkzeug *osmosis*:

```
osmosis --rb file=europe.osm.pbf \
        --bb left=<west> bottom=<süd> right=<ost> top=<nord> \
             clipIncompleteEntities=true \
        --wx file="karte.osm"
```

Download vom *Overpass API* Server:

```
curl --data-urlencode "data=\
[timeout:900];\
(␣node(<süd>,<west>,<nord>,<ost>);\
␣␣<;\
);\
out␣qt;" \
-o karte.osm http://overpass-api.de/api/interpreter
```

Bei Bedarf `timeout` erhöhen

aber: Server hat Sperre gegen zu große Downloads!

Erzeugen der Karte

Zerlegen in Kacheln:

```
java -Xmx2048M -jar splitter.jar karte.osm
```

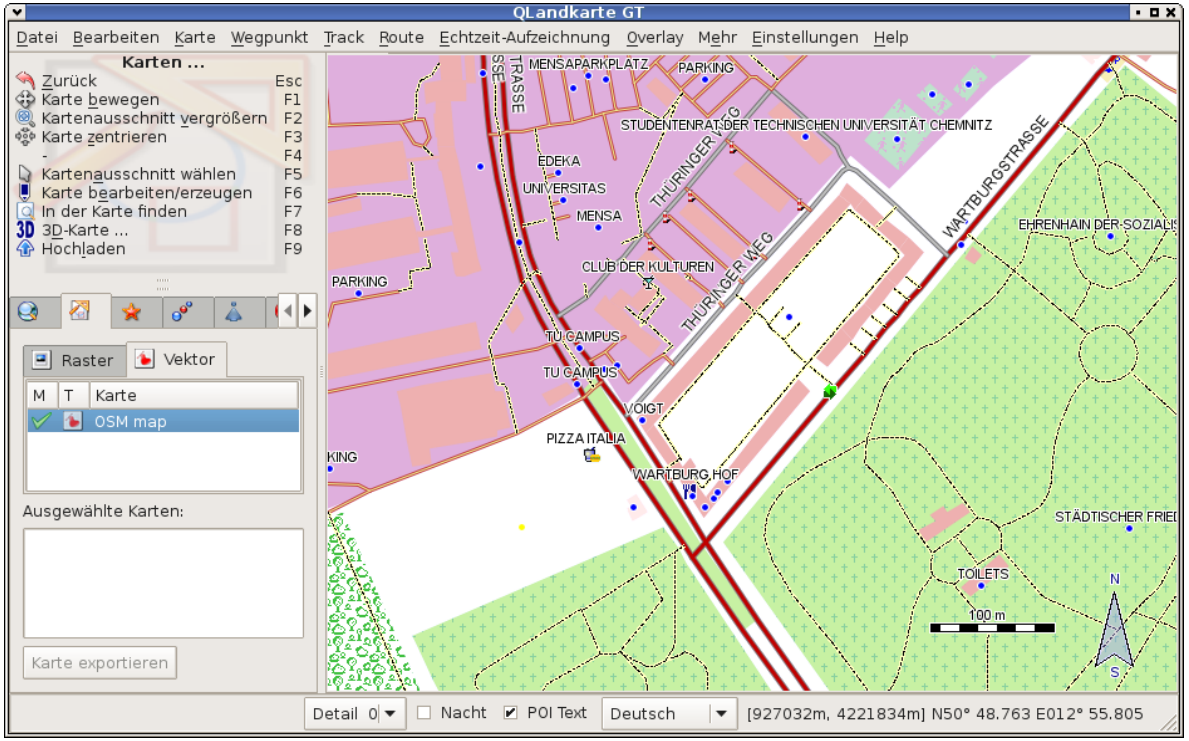
Erzeugen der Garmin-Karte:

```
java -Xmx2048M -jar mkgmap.jar
 --max-jobs \
 --remove-short-arcs --add-pois-to-areas \
 --latin1 --reduce-point-density=10 \
 --product-id=1 \
 --family-id=42 \
 --mapname="Meine␣Karte" \
 --gmapsupp -c template.args --style-file=teddy teddy.typ
```

Family-ID muss zum Typ-File passen!

Test der Karte mit QLandkarteGT

- Neue Karte laden
- Auswahl von `osmmap.tdb`
- Auswahl von `gmapsupp.img`
- Karte zentrieren



Installation der Karte

- Garmin-Gerät in USB-Massenspeichermodus umschalten oder SD-Karte direkt mit Computer verbinden
- Kopieren von `gmapsupp.img` ins Verzeichnis `garmin` der SD-Karte

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

Anforderungen

- Android-Gerät mit GPS-Empfänger (Funkortung zu ungenau)
- Offline-Betrieb möglich (nicht überall WLAN/GPRS)
- freie Karten
- freie Apps

Angebot

Übersicht im OSM-Wiki

Navit

- Vektordaten
- Routing möglich
- sehr zäh

Big Planet Tracks

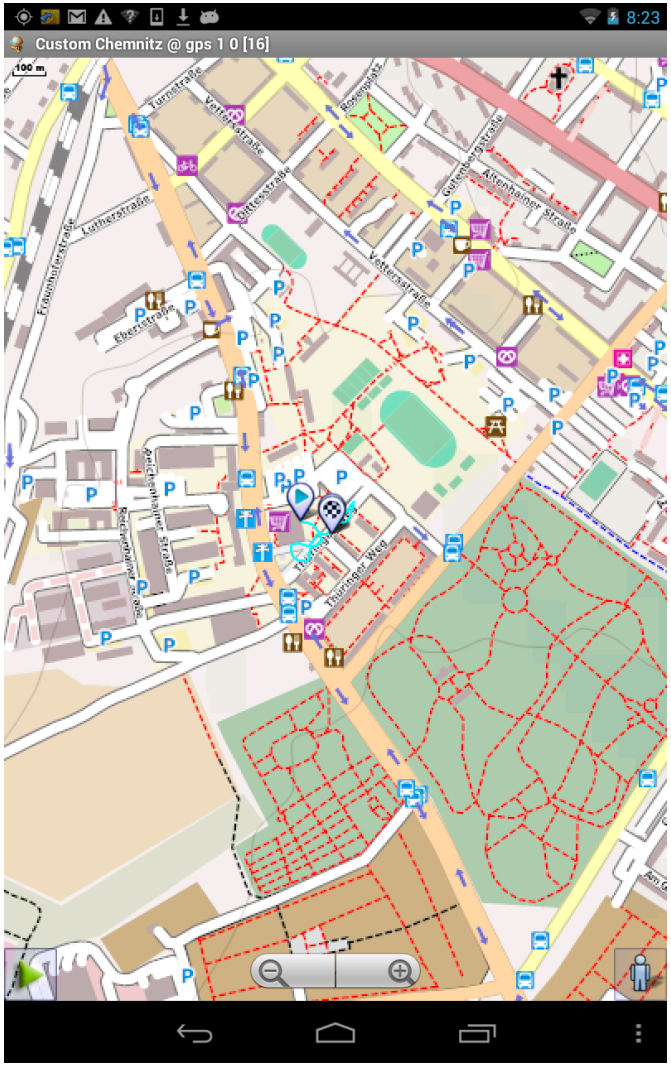
- Kacheln in lokaler Datenbank
- Trackaufzeichnung
- kein Routing

OSMand

- frei, aber nicht Open Source
- ohne Google-Account kein Download
- freie Version auf 3 Karten beschränkt
- Werbungseinblendung!
- $\Rightarrow$ nicht weiter betrachtet

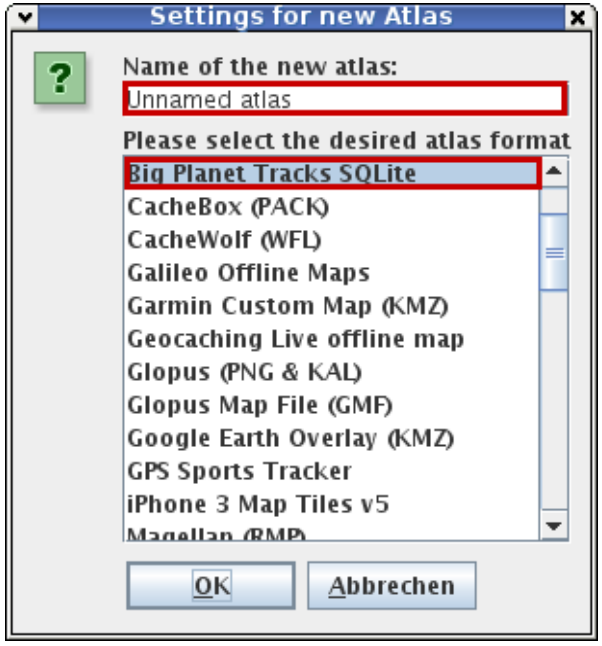
Big Planet Tracks

- Online-Karte (mit Netzverbindung)
- Offline-Karten
- verschiedene Karten wählbar
- Trackdarstellung und -aufzeichnung
- Installation direkt von Webseite möglich
- Offline-Karten mit *Mobile Atlas Creator* (MOBAC)



Mobile Atlas Creator 1

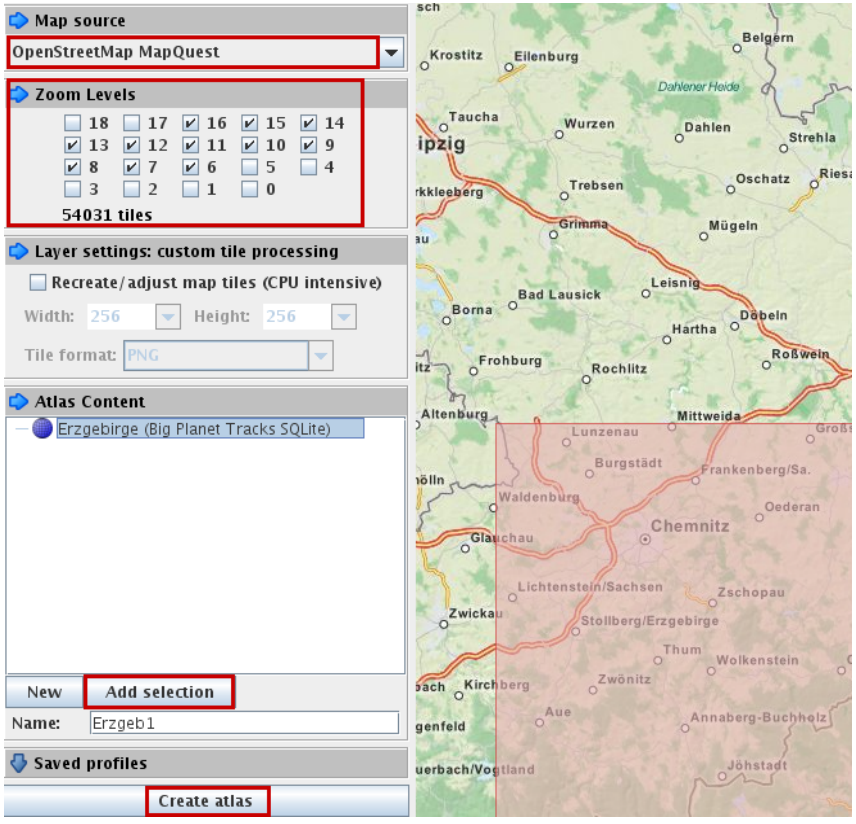
- Java-Anwendung
- Start: `java -jar \`
  Pfad/`Mobile_Atlas_Creator.jar`
- *Atlas→New Atlas*
  Name setzen, Format *Big Planet Tracks SQLite*



Mobile Atlas Creator 2

- Map Source wählen
- Zoom Level wählen
- Gebiet wählen und *Add Selection*
- Aufruf *Create Atlas*

Achtung: Generieren kann mehrere Stunden dauern!



Ergebnis: *SQLite3*-Datenbank
Datenbankstruktur: siehe Beitrag Tagungsband

Übertragen der Karte

- Karte muss auf die SD-Card ins Verzeichnis `RMaps/maps`
- Falls nur MTP-Mode über USB unterstützt:
  Dateisystem `mtpfs` installieren
  Gerät manuell mounten
  Anleitung s. Bibliographie
- Alternative:
  Im Debug-Modus mit *Android Debug Bridge* übertragen:
  `adb push karte.sqlitedb /sdcard/RMaps/maps`

1 Überblick

2 Von der Kugel zur Fläche – Projektionen

3 Kacheln und Slippy-Map

4 Karten für den Druck

5 Online-Karten

6 Karten für Garmin-GPS-Empfänger

7 Karten für Android-Geräte

8 Quellen

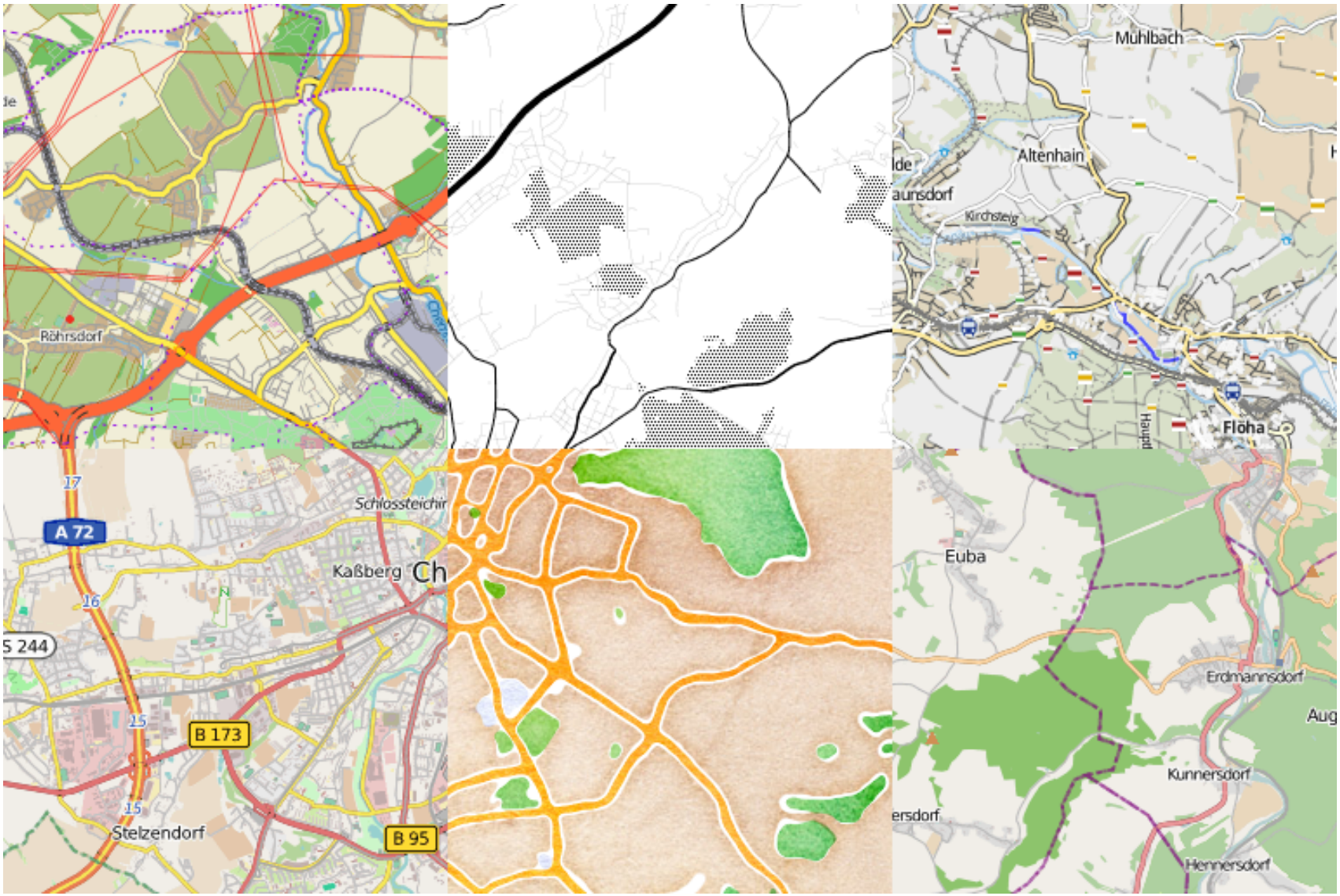
Quellen und weitere Informationen I

- http://wiki.openstreetmap.de – OpenStreetMap Wiki.
- Ramm, Topf: *OpenStreetMap. Die freie Weltkarte nutzen und gestalten.* 3. Auflage 2010. Lehmanns New Media.
- Robinson et al: *Elements of Cartography.* 6th ed. 1995. John Wiley & Sons.
- https://addons.mozilla.org/de/firefox/addon/live-http-headers/ – Live HTTP Headers
- http://ruessel.in-chemnitz.de/osm/bigmap.html – Python-Script bigmap.py.
- http://ruessel.in-chemnitz.de/osm/qbigmap – QBigMap.
- http://www.openlayers.org – JavaScript-Bibliothek OpenLayers.
- http://leafletjs.com – JavaScript-Bibliothek Leaflet.

Quellen und weitere Informationen II

- http://code.google.com/p/big-planet-tracks/ – Android-App Big Planet Tracks.
- http://mobac.sourceforge.net/ – Mobile Atlas Creator.
- http://www.omgubuntu.co.uk/2011/12/how-to-connect-your-android-ice-cream-sandwich-phone-to-ubuntu-for-file-access – MTP-Treiber für Debian.
- http://download.geofabrik.de/ – Download von OpenStreetMap-Vektordaten.
- http://overpass-api.de – Overpass API.
- http://wiki.openstreetmap.org/wiki/User:Computerteddy – Computerteddys Typ- und Styledateien für Garmin-Geräte.
- http://www.mkgmap.org.uk/ – mkgmap.
- http://ruessel.in-chemnitz.de/osm/clt2013 – Webseite zum Vortrag.

Fragen?



Mit `bigmap.py` im Random-Mode generiert